

An Empirical Study of Maintainability in Aspect-Oriented System Evolution Using Coupling Metrics

Haihao Shen, Sai Zhang, Jianjun Zhao
School of Software
Shanghai Jiao Tong University
800 Dongchuan Road, Shanghai 200240, China
{haihaoshen, saizhang, zhao-jj}@sjtu.edu.cn

Abstract

In this paper, we propose a fine-grained coupling metrics suite for aspect-oriented (AO) systems, to measure software changes during system evolution. We also present a correlation model in terms of intermediate processes, for better evaluating the relation between coupling metrics and system maintainability. To investigate the practicability of our proposed model, we have implemented a coupling metrics analysis tool called AJMetrics and performed an empirical study on eight AspectJ benchmarks. The experiment result suggests that our correlation model provides useful information to evaluate the maintainability of AO systems.

1 Introduction

Coupling is an indication of the strength of interconnections between the components in a system [10]. Normally, writing modules with low coupling is thought to be a desirable goal during software development. Coupling metrics are often used as predictors of software external quality attributes such as maintainability, reusability, and reliability. During system evolution, the increasing costs of maintenance have become a major concern for both developers and users. For a typical software system, which continuously evolves to meet ever-changing user needs, maintenance tasks are inevitable during its life cycle.

Aspect-oriented programming (AOP) [5] has been proposed as a brand new programming technique in order to overcome the weakness of object-oriented programming (OOP) when modelling the crosscutting concerns. However, the unique feature of AOP complicates the software analysis.

Although there are some related work on the area of coupling metrics [10, 4, 3] and maintainability [6, 7], the relation between coupling metrics and system maintainability

in AO system is still missing. Thus, how to evaluate the maintainability of AO systems still need to be further investigated.

In this paper, we propose a fine-grained coupling metrics suite for AO programs, to measure software changes during system evolution. We also present a correlation model in terms of *intermediate processes*, for better evaluating the relation between coupling metrics and software maintainability. To investigate the practicability of our proposed model, we have implemented a coupling metrics analysis tool called AJMetrics and performed an empirical study on eight AspectJ benchmarks. The experiment result suggests that our correlation model provides useful information to evaluate the maintainability of AO systems.

The rest of this paper is organized as follows. Section 2 introduces the background of this paper. Section 3 proposes a suite of fine-grained coupling metrics and Section 4 presents a correlation model between coupling metrics and system maintainability. Section 5 performs the empirical study on eight AspectJ benchmarks and discusses the effectiveness of the correlation model. Concluding remarks are given in Section 6.

2 Background

We next introduce the background on several aspects in AO systems including terminology and software changes before we formally propose a correlation model between coupling metrics and maintainability.

2.1 Terminology

To define a fine-grained coupling metrics suite better, we reuse a terminology [10] for an AO system. AO system considered in this paper is composed of *aspects* and *classes*. We do not consider *interfaces* specifically because they can be handled similarly with *classes*.

Definition 1 (AO System) An AO system S consists of a set of aspects, $A(S)$ and a set of classes, $C(S)$.

Definition 2 (Ancestors of an Aspect) Let S be an AO system. For each aspect $a \in A(S)$, let $Ancestors(a) \subset A(S)$ be the set of ancestor aspects of a .

Definition 3 (Ancestors of a Class) Let S be an AO system. For each class $c \in C(S)$, let $Ancestors(c) \subset C(S)$ be the set of ancestor classes of c .

Definition 4 (Modules of an Aspect) Let S be an AO system. For each aspect $a \in A(S)$, let $\mathcal{A}(a)$ be the set of advices of a , $\mathcal{I}(a)$ be the set of intertype declarations of a , $\mathcal{P}(a)$ be the set of pointcuts of a , and $\mathcal{M}(a)$ be the set of methods of a .

Definition 5 (Modules of a Class) Let S be an AO system. For each class $c \in C(S)$, let $\mathcal{M}(c)$ be the set of methods of c .

Definition 6 (Attributes of Aspects and Classes) Let S be an AO system. For each $a \in A(S)$, let $Attributes(a)$ be the set of attributes of aspect a . For each $c \in C(S)$, let $Attributes(c)$ be the set of attributes of class c .

2.2 Software Changes

To reflect the maintenance quantitatively, we consider software changes as the indicators of maintenance tasks. Here, we reuse the definition of atomic changes [9] and choose some of them as software changes in this paper.

Table 1. Software Changes in Aspect Code

| Change | Definition for Change | Change | Definition for Change |
|--------|------------------------|--------|--|
| AA | Add an Empty Aspect | DIF | Delete an Introduced Field |
| DA | Delete an Empty Aspect | AHD | Add a Hierarchy Declaration |
| AEA | Add an Empty Advice | DHD | Delete a Hierarchy Declaration |
| DEA | Delete an Empty Advice | CIFI | Change an Introduced Field Initializer |
| CAB | Change an Advice Body | INM | Introduce a New Method |
| ANP | Add an New Pointcut | DIM | Delete an Introduced Method |
| CPB | Change a Pointcut Body | CIMB | Change an Introduced Method Body |
| DPC | Delete a Pointcut Body | ASED | Add a Soften Exception Declaration |
| INF | Introduce a New Field | DSED | Delete a Soften Exception Declaration |

Table 2. Software Changes in Base Code

| Change | Definition for Change | Change | Definition for Change |
|--------|-----------------------|--------|------------------------|
| AC | Add an Empty Class | AM | Add an Empty Method |
| DC | Delete an Empty Class | DM | Delete an Empty Method |
| AF | Add a Field | CM | Change Body of Method |
| DF | Delete a Field | | |

3 A Suite of Fine-grained Coupling Metrics

Several metrics suites have been proposed [3, 4], however, most of them concentrate on coarse-grained coupling

metrics. In this section, we present a suite of fine-grained coupling metrics, which focus on the module dependencies in an AO system. This fine-grained coupling metrics suite listed in Table 3 based on the coupling framework carried out by Ceccato [4] including CFA, CMC, RFM, CAE, and CDA.

4 Correlation Model between Coupling Metrics and Maintainability

We next present a correlation model between coupling metrics and maintainability for AO systems in terms of *intermediate processes* consisting of the relation between software changes and maintenance tasks, and the relation between maintenance tasks and maintainability.

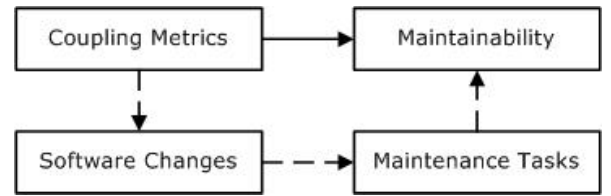


Figure 1. Correlation Model

In Figure 1, the direction of the three dotted lines indicates *intermediate processes* when building the relation between coupling metrics and maintainability, which is shown as the direction of the solid line. Before introducing our correlation model, we first propose two assumptions for intermediate processes:

Assumption 1 (Relation between software changes and maintenance tasks): More software changes occur, more maintenance tasks maintain staffs are likely to do during system evolution, and vice versa.

Assumption 2 (Relation between maintenance tasks and maintainability): More maintenance tasks maintain staffs do, less maintainability system is likely to reveal during system evolution, and vice versa.

We next redefine a suite of metrics for changes, based on the fine-grained coupling metrics. Our redefined coupling metrics for changes (listed in Table 4) can be used to measure fine-grained coupling of changes during system evolution.

Definition 7 (Maintenance Task) Let S be an original AO system, S' be a new AO system and let c be a kind of corrective change from S to S' . Here, maintenance task reflects the number of software changes occurred from S to S' .

We next define a list of maintenance tasks in Table 5 and classify them into two parts: *individual maintenance task* and *global maintenance task*. Individual maintenance task includes E1 – E7 while global maintenance task is E8.

Table 3. A Suite of Fine-grained Coupling Metrics

| Metric | Definition |
|-------------------|--|
| CFA(a, f) | The number of modules accessing a filed $f \in \text{Attributes}(a)$ where $a \in A(S)$ |
| CMC(a, m) | The number of modules calling a method $m \in \mathcal{M}(a)$ where $a \in A(S)$ |
| RFM(a, m) | The number of methods potentially executed in response to message of a method $m \in \mathcal{M}(a)$ received by a given aspect a where $a \in A(S)$ |
| CAM(a, m) | The number of aspects containing advices possibly triggered by the execution of the method $m \in \mathcal{M}(a)$ where $a \in A(S)$ |
| CIM(a, p) | The number of modules explicitly named in the pointcut $p \in \mathcal{P}(a)$ belonging to a given aspect a where $a \in A(S)$ |
| CDP(a, p) | The number of modules affected by the pointcut $p \in \mathcal{P}(a)$ in a given aspect a where $a \in A(S)$ |
| CAA(a, α) | The number of aspects containing advices possibly triggered by the execution of advice $\alpha \in \mathcal{A}(a)$ where $a \in A(S)$ |
| CDI(a, i) | The number of modules affected by the intertype declaration $i \in \mathcal{I}(a)$ where $a \in A(S)$ |
| CAI(a, i) | The number of aspects containing advices possibly triggered by the execution of intertype declaration $i \in \mathcal{I}(a)$ where $a \in A(S)$ |
| CFA(c, f) | The number of modules accessing a filed $f \in \text{Attributes}(c)$ where $c \in C(S)$ |
| CMC(c, m) | The number of modules calling a method $m \in \mathcal{M}(c)$ where $c \in C(S)$ |
| RFM(c, m) | The number of methods potentially executed in response to message of a method $m \in \mathcal{M}(c)$ received by a given class c where $c \in C(S)$ |

Table 4. Coupling Metrics for Changes

| Metric | Definition for Metric |
|--------|---|
| M1 | $ \sum_{a \in A(S)} \text{Ancestors}(a) - \sum_{a' \in A(S')} \text{Ancestors}(a') $ |
| M2 | $ \sum_{a \in A(S)} \sum_{f \in \text{Attributes}(a)} CFA(a, f) - \sum_{a' \in A(S')} \sum_{f' \in \text{Attributes}(a')} CFA(a', f') $ |
| M3 | $ \sum_{a \in A(S)} \sum_{m \in \mathcal{M}(a)} CMC(a, m) - \sum_{a' \in A(S')} \sum_{m' \in \mathcal{M}(a')} CMC(a', m') $ |
| M4 | $ \sum_{a \in A(S)} \sum_{m \in \mathcal{M}(a)} CAM(a, m) - \sum_{a' \in A(S')} \sum_{m' \in \mathcal{M}(a')} CAM(a', m') $ |
| M5 | $ \sum_{a \in A(S)} \sum_{m \in \mathcal{M}(a)} RFM(a, m) - \sum_{a' \in A(S')} \sum_{m' \in \mathcal{M}(a')} RFM(a', m') $ |
| M6 | $ \sum_{a \in A(S)} \sum_{p \in \mathcal{P}(a)} CIM(a, p) - \sum_{a' \in A(S')} \sum_{p' \in \mathcal{P}(a')} CIM(a', p') $ |
| M7 | $ \sum_{a \in A(S)} \sum_{p \in \mathcal{P}(a)} CDP(a, p) - \sum_{a' \in A(S')} \sum_{p' \in \mathcal{P}(a')} CDP(a', p') $ |
| M8 | $ \sum_{a \in A(S)} \sum_{\alpha \in \mathcal{A}(a)} CAA(a, \alpha) - \sum_{a' \in A(S')} \sum_{\alpha' \in \mathcal{A}(a')} CAA(a', \alpha') $ |
| M9 | $ \sum_{a \in A(S)} \sum_{i \in \mathcal{I}(a)} CDI(a, i) - \sum_{a' \in A(S')} \sum_{i' \in \mathcal{I}(a')} CDI(a', i') $ |
| M10 | $ \sum_{a \in A(S)} \sum_{i \in \mathcal{I}(a)} CAI(a, i) - \sum_{a' \in A(S')} \sum_{i' \in \mathcal{I}(a')} CAI(a', i') $ |
| M11 | $ \sum_{c \in C(S)} \text{Ancestors}(c) - \sum_{c' \in C(S')} \text{Ancestors}(c') $ |
| M12 | $ \sum_{c \in C(S)} \sum_{f \in \text{Attributes}(c)} CFA(c, f) - \sum_{c' \in C(S')} \sum_{f' \in \text{Attributes}(c')} CFA(c', f') $ |
| M13 | $ \sum_{c \in C(S)} \sum_{m \in \mathcal{M}(c)} CMC(c, m) - \sum_{c' \in C(S')} \sum_{m' \in \mathcal{M}(c')} CMC(c', m') $ |
| M14 | $ \sum_{c \in C(S)} \sum_{m \in \mathcal{M}(c)} RFM(c, m) - \sum_{c' \in C(S')} \sum_{m' \in \mathcal{M}(c')} RFM(c', m') $ |

Table 5. List of Maintenance Tasks

| Maintenance Task | Definition for Maintenance Task |
|------------------|--|
| E1 | The number of change c whose kind is AA or DA from S to S' |
| E2 | The number of change c whose kind is INF, DIF, CIFI, INM, DIM or CIMB from S to S' |
| E3 | The number of change c whose kind is AEA, DEA or CAB from S to S' |
| E4 | The number of change c whose kind is ANP, CPB, CDPC, ASER or DSED from S to S' |
| E5 | The number of change c whose kind is AC, DC, AHD or DHD from S to S' |
| E6 | The number of change c whose kind is AF or DF from S to S' |
| E7 | The number of change c whose kind is AM, DM or CM from S to S' |
| E8 | The sum of E_i where i is from 1 to 7 |

Our experience in system maintainability has motivated us that, if a change occur between two software versions, the maintenance task result from the related corrective change should be identified. If more maintenance tasks need to be carried out by maintain staffs, the system maintainability will be worse.

Definition 8 (Correlation Degree) Let S be an original AO system and S' be a new AO system. Let M be a coupling metric and E be a maintenance task from S to S' , the correlation degree between M and E is denoted as *Correla-*

tion (M, E).

We next define a suite of correlation indicators between coupling metrics and maintenance tasks, which could be used to reflect the correlation degree between coupling metrics and maintainability quantitatively.

Table 6. Correlation Indicators

| Correlation Name | Definition | Granularity |
|------------------|----------------------------|-------------|
| CDI_1 | Correlation (M_1, E_1) | Aspect (I) |
| CDI_2 | Correlation (M_2, E_6) | Field (I) |
| CDG_1 | Correlation (M_1, E_8) | Aspect (G) |
| CDG_2 | Correlation (M_2, E_8) | Field (G) |

5 Empirical Evaluation

To investigate the practicability of our proposed model, we implement a coupling metrics analysis tool called AJMetrics [8] and perform an empirical study on eight AspectJ benchmarks. We next discuss whether our correlation model can provide helpful information to evaluate the maintainability of an AO system.

5.1 Subject Programs

We use eight AspectJ benchmarks shown in Table 7 for the experimental study. The first three and the spacewar ex-

ample are included in the AspectJ compiler example package. The remaining programs are obtained from the abc benchmark package [2].

Table 7. Subject Programs

| Programs | Line of Code | Version | Method |
|-----------|--------------|---------|--------|
| QuickSort | 111 | 3 | 18 |
| Figure | 147 | 4 | 23 |
| Bean | 199 | 3 | 12 |
| Tracing | 1059 | 4 | 44 |
| NullCheck | 2991 | 4 | 196 |
| Lod | 3075 | 2 | 220 |
| Dcm | 3423 | 2 | 249 |
| Spacewar | 3053 | 2 | 288 |

5.2 Experiment Results

For each AspectJ benchmark, AJMetrics automatically compute the metrics values. In our experiment, an analysis tool called SPSS [1], which provides statistical analysis of data, is used to analyze the correlation degree between coupling metrics and maintenance tasks by means of Spearman analysis.

Table 8. Results for Correlation Indicators

| Correlation Degree | P-Value | Sig | Correlation Degree | P-Value | Sig |
|--------------------|---------|-------|--------------------|---------|-------|
| CDI_{14} | 0.892 | 0.000 | CDI_5 | 0.212 | 0.430 |
| CDI_{13} | 0.787 | 0.000 | CDI_2 | 0.182 | 0.501 |
| CDI_{12} | 0.703 | 0.002 | CDG_7 | 0.178 | 0.509 |
| CDG_{14} | 0.691 | 0.003 | CDG_8 | 0.178 | 0.509 |
| CDG_{12} | 0.647 | 0.007 | CDG_2 | 0.157 | 0.563 |
| CDI_{11} | 0.613 | 0.012 | CDG_9 | 0.155 | 0.567 |
| CDG_{13} | 0.542 | 0.030 | CDG_{10} | 0.155 | 0.567 |
| CDI_9 | 0.424 | 0.101 | CDG_3 | 0.098 | 0.719 |
| CDI_{10} | 0.424 | 0.101 | CDI_7 | -0.122 | 0.653 |
| CDG_{11} | 0.397 | 0.128 | CDI_8 | -0.461 | 0.072 |
| CDI_1 | 0.342 | 0.194 | CDI_4 | Null | Null |
| CDG_1 | 0.328 | 0.215 | CDI_6 | Null | Null |
| CDG_5 | 0.281 | 0.292 | CDG_4 | Null | Null |
| CDI_3 | 0.226 | 0.400 | CDG_6 | Null | Null |

5.3. Correlation Analysis

After analyzing the correlation degree between coupling metrics and maintenance tasks, we find several interesting conclusions as follows:

- Coupling metrics caused by software changes in the base code are always more correlated with maintainability than those caused by changes in the aspect code.
- Correlation degree between coupling metrics and global maintenance tasks is always lower than correlation degree between coupling metrics and individual maintenance tasks.
- Among all the correlation indicators between coupling metrics and individual maintenance tasks caused by software changes in the aspect code, different features in AspectJ have different effect on maintenance tasks.

6 Concluding Remarks

In this paper, we proposed a fine-grained coupling metrics suite for AO systems. Based on this coupling metrics suite, we measured coupling metrics for software changes during system evolution. We also presented a correlation model between coupling metrics and system maintainability. The experiment result suggests that our correlation model provides useful information to evaluate the maintainability of an AO system.

In our future work, we would like to perform further empirical evaluations on larger AO systems. We will also investigate more applications of our metrics suite and extend the existing suite to predict system maintainability.

Acknowledgments

This work was supported in part by National High Technology Development Program of China (Grant No. 2006AA01Z158), National Natural Science Foundation of China (NSFC) (Grant No. 60673120), and Shanghai Pujiang Program (Grant No. 07pj14058). We would like to thank Hanyue Yang for her discussion on this work.

References

- [1] SPSS. <http://www.spss.com/>.
- [2] The AspectBench Compiler. <http://abc.comlab.ox.ac.uk/>.
- [3] M. Bartsch and R. Harrison. Towards an empirical validation of aspect-oriented coupling metrics. In *ASAT Workshop at the Aspect-Oriented Developing Conference (AOSD)*, Vancouver, BC, 2007.
- [4] M. Ceccato and P. Tonella. Measuring the Effects of Software Aspectization. *Proceedings of the 1st Workshop on Aspect Reverse Engineering (CD-ROM)*, The Netherlands, 2004.
- [5] K. e. a. Gregor. Aspect-oriented programming. In *11th European Conference on Object-Oriented Programming*, pages 220–242, 1997.
- [6] W. Li and S. Henry. Object-oriented metrics that predict maintainability. *Journal of Systems and Software*, 23(2):111–122, 1993.
- [7] C. Rajaraman and M. Lyu. Reliability and maintainability related software coupling metrics in C++ programs. *Software Reliability Engineering, 1992. Proceedings., Third International Symposium on*, pages 303–311, 1992.
- [8] H. Shen and J. Zhao. An evaluation of coupling metrics for aspect-oriented software. Technical Report SJTU-CSE-TR-07-04, Center for Software Engineering, SJTU, Dec 2007.
- [9] S. Zhang and J. Zhao. Change impact analysis for AspectJ programs. Technical Report SJTU-CSE-TR-07-01, Center for Software Engineering, SJTU, Jan 2007.
- [10] J. Zhao. Measuring coupling in aspect-oriented system. In *10th International Software Metrics Symposium (METRICS'2004), (Late Breaking Paper)*, Chicago, USA, Sept. 14–16 2004.